UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD INGENIERÍA MECÁNICA ELCTRICA

"APLICACIÓN DEL MICROCONTROLADOR M68HC12B PARA EL CONTROL DE UN ROBOT DE 4 GRADOS DE LIBERTAD"

COMO REQUISITO PARA OPTAR AL TITULO DE MASTER EN INGENIERIA CON ESPECIALIZACIÓN EN MECATRONICA

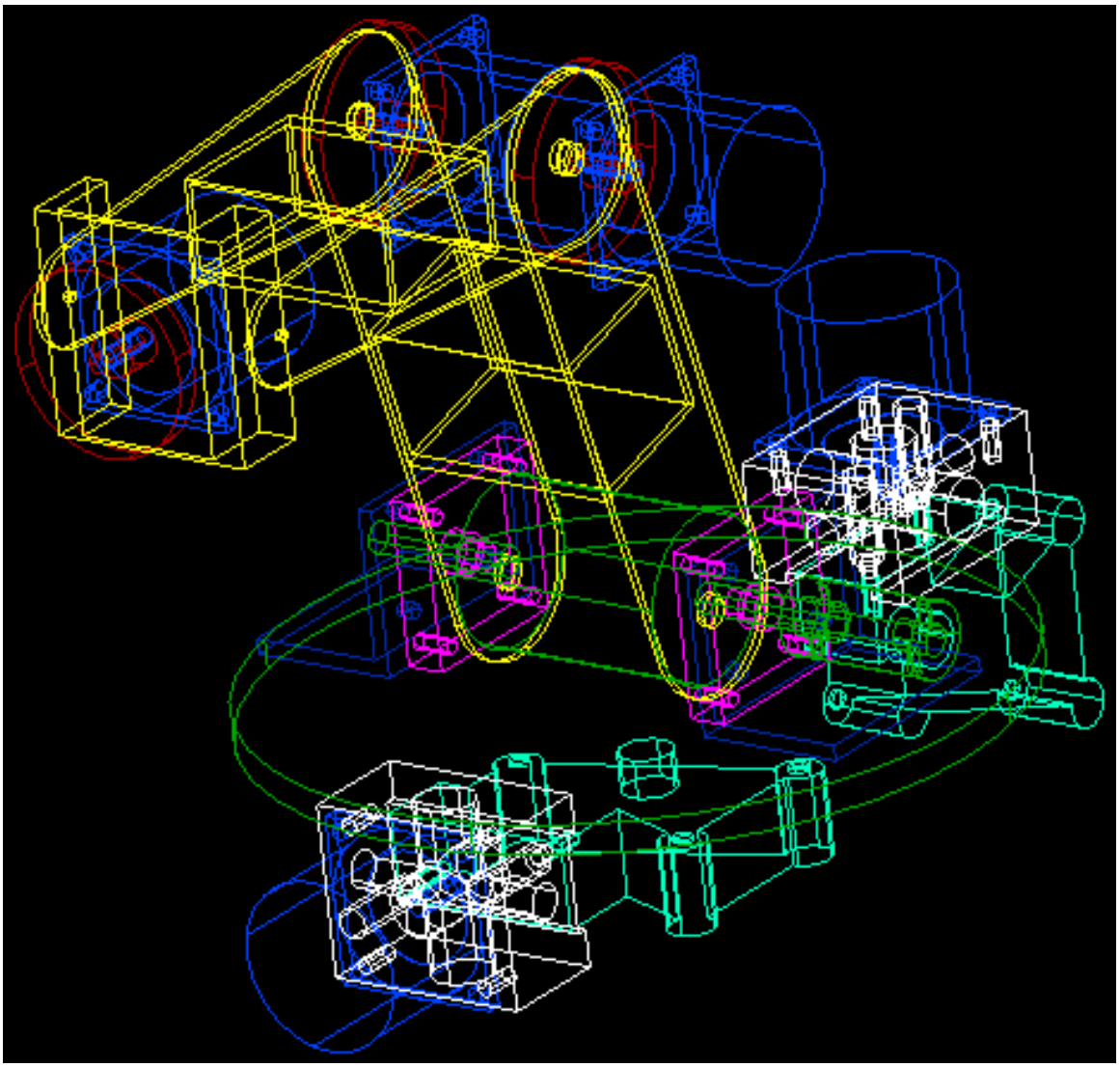
PRESENTA

MAURICIO HERNÁNDEZ PEREZ

San Nicolás de los Garza, N.L. Diciembre de 2007

Aplicación del microcontrolador M68HC12B para el control de un robot de 4 grados de libertad.

ROBOTICA



La robótica es una ciencia aplicada que ha sido considerada como una combinación de tecnología de las maquinas-herramienta y de la informática. Comprende campos tan aparentemente diferentes como diseño de maquinas, teoría de control, microelectrónica, programación de computadoras, inteligencia artificial y teoría de la producción.

Para poder apreciar la tecnología de la robótica y su programación debe conocerse la forma en que los robots se aplican en la industria.
los robots son sistemas mecatronicos en los cuales interactúan 3 componentes principales:
la parte mecánica
la parte electrónica
la programación

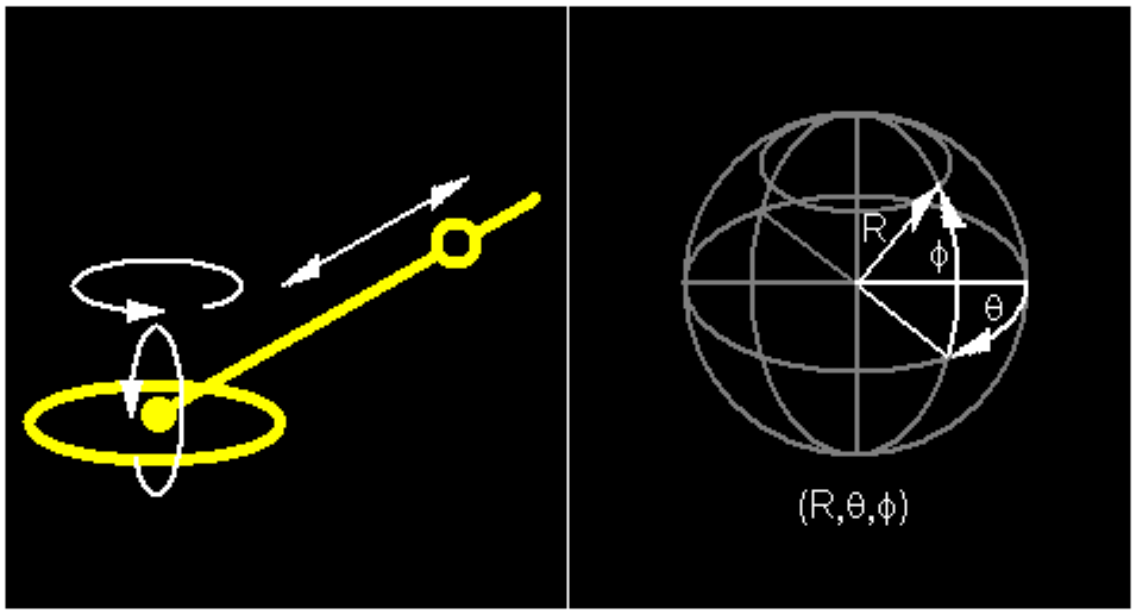
ANATOMIA DEL ROBOT

La construcción física del robot se divide en cuerpo (brazo) muñeca y efector final

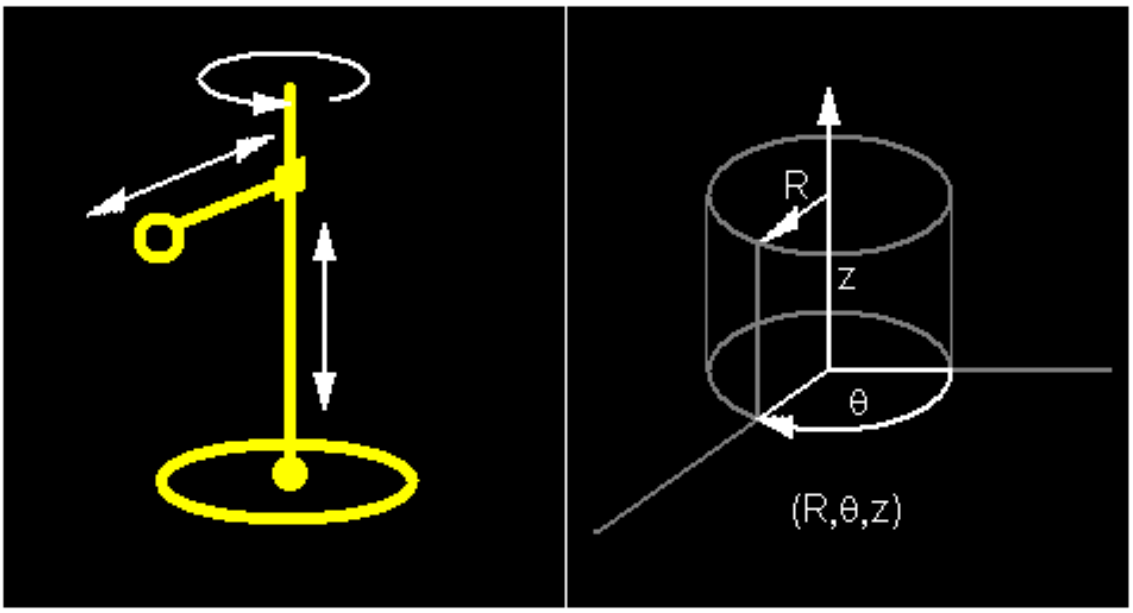
CUERPO

La mayoría de los robots comercialmente disponibles en la actualidad tienen una de estas 4 configuraciones básicas:

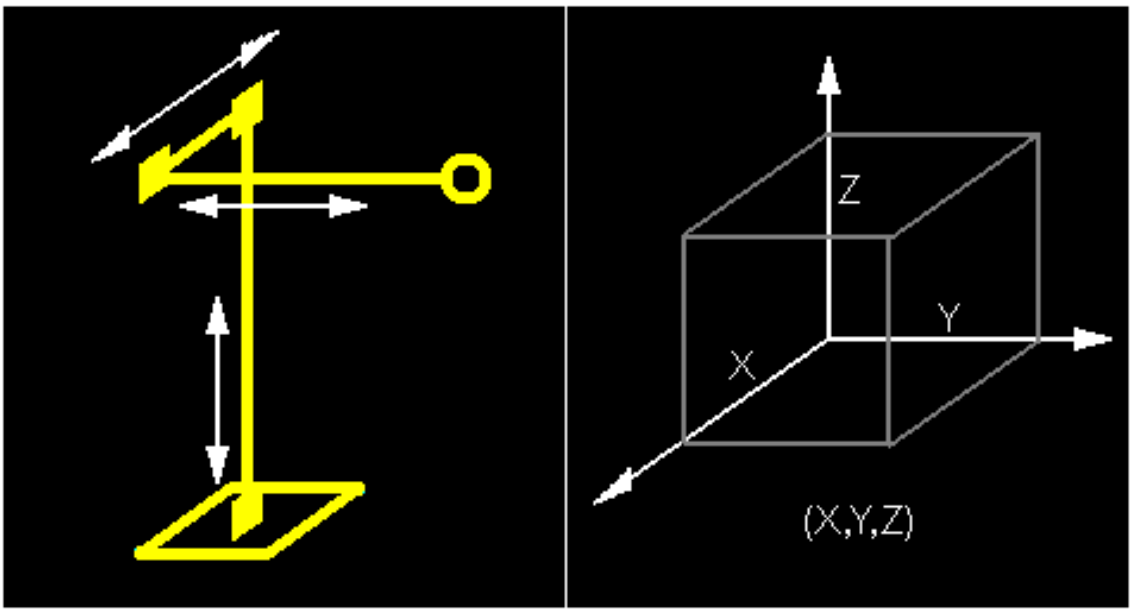
La **configuración polar** utiliza un brazo telescopio que puede elevarse o bajar alrededor de un pivote horizontal. Este pivote esta montado sobre una base giratoria. Estas articulaciones proporcionan al robot la capacidad para desplazar su brazo dentro de un espacio esférico y de aquí la denominación de robot “coordenadas esféricas” pues el señalamiento de cada punto dentro del volumen de trabajo por medio de sus articulaciones se asemeja a la forma en que se describiría un punto en el espacio usando coordenadas esféricas.



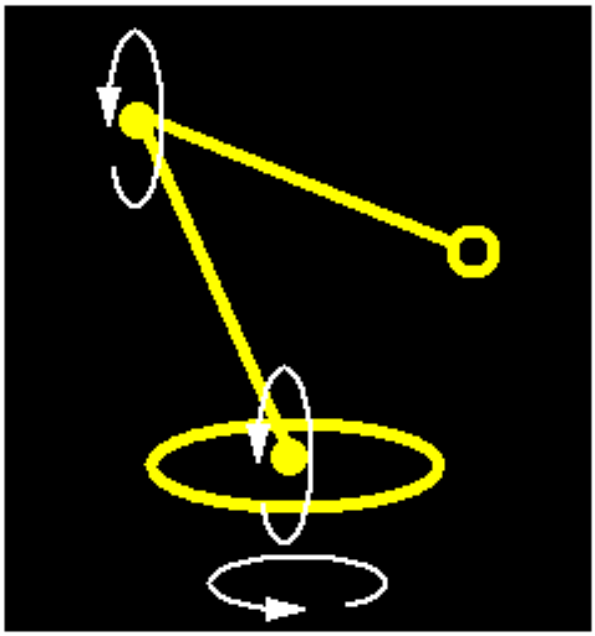
La **configuración cilíndrica** utiliza una columna vertical y un dispositivo de desplazamiento que puede moverse hacia arriba o hacia abajo a lo largo de la columna. El brazo esta unido a un dispositivo deslizante de modo que puede moverse en sentido radial. Haciendo girar la columna el robot es capaz de conseguir un volumen de trabajo que se aproxima a un cilindro.



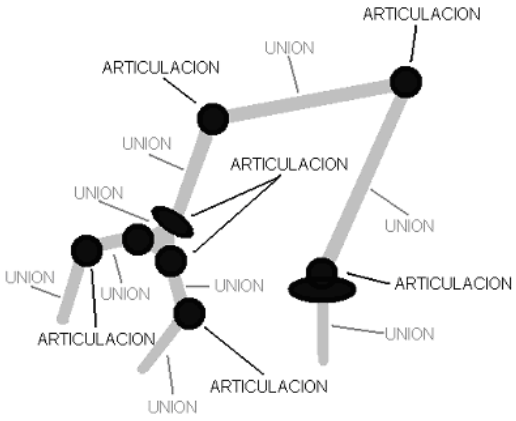
El robot de **coordenadas cartesianas**, utiliza 3 dispositivos deslizantes perpendiculares para construir los ejes X, Y, y Z. El volumen de trabajo que describen esta articulaciones es aproximada a un cubo.



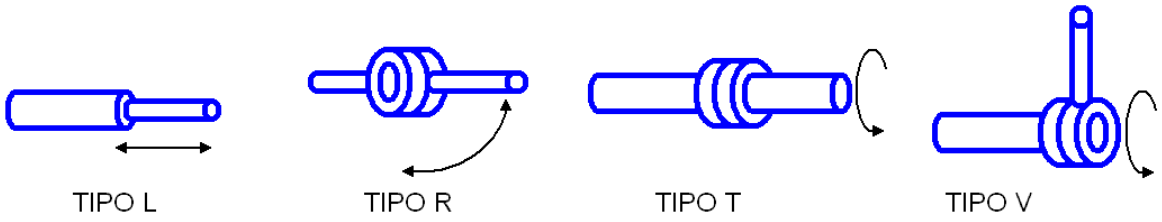
El robot de brazo articulado esta constituido por 2 componentes rectos que corresponden al brazo y al antebrazo humanos montados sobre un pedestal vertical. Estos componentes están conectados por 2 articulaciones que corresponden al hombro y al codo.



Sin importar cual es la configuración anatómica del robot siempre estará constituido por dos componentes elementales: unión y articulación.



La unión es el elemento rígido del cuerpo del robot.
La articulación es la conexión entre al menos 2 uniones y determinan uno o mas grados de libertad. Dependiendo de movimiento relativo que hay entre la unión de entrada y la unión de salida, las articulaciones se clasifican en:



ARTICULACIÓN TIPO L o lineal: permite desplazamiento paralelo y lineal de la unión de salida. Es muy usada en robots cartesianos.
ARTICULACIÓN TIPO R o rotacional: esta articulación permite movimiento angular longitudinal en la unión de salida. Es muy usada en brazos articulados
ARTICULACIÓN TIPO T o torsión: esta articulación permite movimiento angular transversal en la unión de salida. Se usa comúnmente en la muñeca del robot.
ARTICULACIÓN TIPO V o revolución: permite el movimiento angular ortogonal entre la unión de entrada y la unión de salida.

MUÑECA

la muñeca esta constituida por varios componentes que permiten orientarse en una diversidad de posiciones. Es la parte que une al cuerpo del robot con el efector final.

EFECTOR FINAL

los efectores finales pueden dividirse en 2 categorías: pinzas y herramientas. Las pinzas se utilizaran para agarrar un objeto, normalmente la pieza de trabajo, y sujetarlo durante el ciclo de trabajo del robot. Hay una diversidad de metidos de sujeción que pueden utilizarse, además de los medios mecánicos obvios de agarrar la pieza entre 2 o mas dedos. Estos métodos suplementarios incluyen el empleo de casquetes de sujeción, imanes ganchos y cucharas. Una herramienta se utilizaría como efector final en aplicaciones en donde exijan al robot realizar alguna operación en la pieza de trabajo. Estas aplicaciones incluyen soldadura por puntos soldadura en arco, la pintura por pulverización y las operaciones de taladrado. En cada caso, la herramienta particular esta unida ala muñeca del robot para realizar la operación.

impulsores

la capacidad del robot para desplazar su cuerpo mediante sus articulaciones se proporciona por el sistema de impulsión utilizado para accionar el robot. El sistema impulsor determina la velocidad de los movimientos del brazo la resistencia mecánica y su rendimiento dinámico. En cierta medida, el sistema de impulsor determina las clases de aplicaciones que puede realizar el robot. Los tipos de sistema de impulsión son:
Impulsión hidráulica
Impulsión Eléctrica
Impulsión Neumática

impulsores eléctricos: motores de pasos

Los motores de pasos son motores que eléctricos que se mueven a partir de pulsos.
los motores de pasos son comunmente usados en impresoras, manejadores de discos, alimentadores de papel, plotters, controles en aviones, brazos mecánicos y robots.

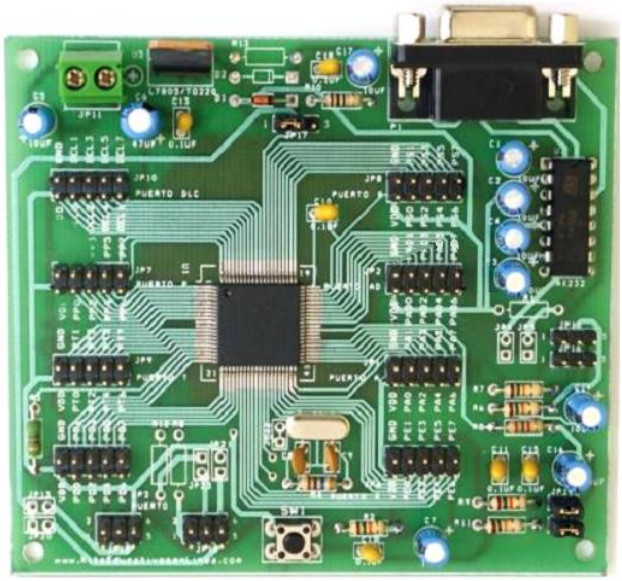
los controles para este tipo de motores se pueden implementar basándose en circuitos TTL, sin embargo los microcontroladores como los PIC son especialmente útiles para este tipo de aplicaciones.

los motores de pasos pueden caer dentro de dos tipos: de imanes permanentes y de reluctancia variable. los motores mas populares son los de imanes permanentes.
los motores de pasos pueden verse como actuadotes electrónicos que mueven su flecha unos cuantos grados, varios pulsos para girar la flecha mas grados. por ejemplo si un los motor de pasos tiene de STEP ANGLE: 7.5 grados, significa que por cada pulso el motor avanza 7.5 grados. claro que para una vuelta completa, es decir 360° se requieren 48 pulsos. entonces, el avance de la flecha del motor esta relacionada con el numero de pulsos y la velocidad de

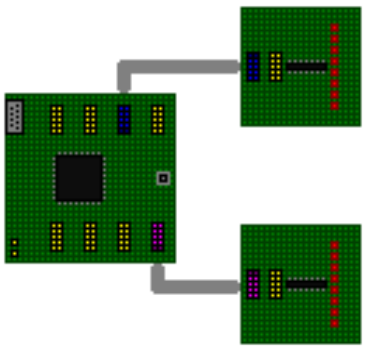
rotación con su frecuencia. entre cada pulso el motor mantiene su posición sin ayuda de embragues.
el motor de pasos no responde en forma directa a los trenes de pulsos. al tener varios devanados, estos deben ser energizados según una secuencia determinada que al invertirse hace que el motor también invierta su sentido de rotación.
los controles para los motores de pasos pueden ser unipolares o bipolares, según si tienen la capacidad de invertir el voltaje de alimentación en los devanados del motor o no.
los controles unipolares son los mas sencillos y también los mas económicos. estos controles solo pueden usarse con motores de 5 o 6 hilos. cada devanado del motor cuenta con un tap central.
los controles bipolares pueden conectarse a motores de 4, 5 o 6 hilos, debido a que tienen la capacidad de invertir el sentido de la corriente que pasa por cada devanado del motor de pasos.
en los controles Full step un pulso hace girar el motor los grados nominales del motor.
en los controles Half step un pulso hace girar el motor la mitad de los grados nominales del motor. por ejemplo si lo grados por paso son 7.5, en este modo de funcionamiento se requieren 96 pulsos para hacer girar el motor una vuelta completa.
En los controles micro step un pulso hace girar la décima parte de los grados nominales del motor.

PROGRAMACION

La tarjeta de control tiene la tarea de coordinar el movimiento de cada uno de los motores. Para este proyecto se utilizo una tarjeta de microcontrolador M68HC12B.



Los puertos A y B del microcontrolador pueden ser declarados como salidas desde el programa. Los pines de los puertos A y B entregan valores binarios de 8 bits (256) los cuales pueden ser utilizados para entregar los pulsos a cada motor de pasos.



puertos A y B del microcontrolador

bobina	1	2	3	4
1er paso	V+	V-	V+	V-
2do paso	V-	V+	V+	V-
3er paso	V-	V+	V-	V+
4to paso	V+	V-	V-	V+

salida	bit3	bit2	bit1	bit0	valor del puerto en exadecimal
1er paso	1	0	1	0	0a
2do paso	0	1	1	0	06
3er paso	0	1	0	1	05
4to paso	1	0	0	1	09

Por tanto cada puerto puede alimentar los pulsos de 2 motores a la ves. Pues se utilizan 4 de los 8 bits disponibles en cada puerto. De este modo, los primeros 4 bits (bit0, bit1,bit2,bit3) del puerto A determinan los pulsos que conforman los pasos para el motor 1 y los últimos 4 bit (bit4, bit5, bit6, bit7) del mismo puerto A determinan los pulsos que conforman los pasos para el motor 2.

pasos para el motor 1

1er paso	puerto A con el valor	*a	(****1010)
2do paso	puerto A con el valor	*6	(****0110)
3er paso	puerto A con el valor	*5	(****0101)
4to paso	puerto A con el valor	*9	(****1001)

pasos para el motor 2

1er paso	puerto A con el valor	a*	(1010****)
2do paso	puerto A con el valor	6*	(0110****)
3er paso	puerto A con el valor	5*	(0110****)
4to paso	puerto A con el valor	9*	(1001****)

pasos para el motor 3

1er paso	puerto B con el valor	*a	(****1010)
2do paso	puerto B con el valor	*6	(****0110)
3er paso	puerto B con el valor	*5	(****0101)
4to paso	puerto B con el valor	*9	(****1001)

pasos para el motor 4

1er paso	puerto B con el valor	a*	(1010****)
2do paso	puerto B con el valor	6*	(0110****)
3er paso	puerto B con el valor	5*	(0110****)
4to paso	puerto B con el valor	9*	(1001****)

donde el valor * representa cualquiera de los valores de los pasos que representan los pulsos del motor correspondiente.

ALGORITMOS PARA EL MICROCONTROLADOR

El primer problema que hay que resolver es el de la entrega de pulsos al motor. Mientras que el motor de pasos recibe voltajes positivos y negativos en sus 4 entradas:

bobina	1	2	3	4
1er paso	+12V	-12V	+12V	-12V
2do paso	-12V	+12V	+12V	-12V
3er paso	-12V	+12V	-12V	+12V
4to paso	+12V	-12V	-12V	+12V

Podemos utilizar la salida digital de alguno de los puertos en este caso puerto A para entregar la siguiente secuencia de números

salida	bit3	bit2	bit1	bit0
1er paso	5V	0V	5V	0V
2do paso	0V	5V	5V	0V
3er paso	0V	5V	0V	5V
4to paso	5V	0V	0V	5V

Puede verse al paso formado por +12V en la bobina 1 -12V en la bobina 2, +12V en la bobina 3, -12V en la bobina 4, como el numero exadecimal "a" (1010 en binario) siempre y cuando el arreglo de las interfaces entre dicho puerto y el motor de pasos haga que, un 1 logico (+5V) en los pines del puerto de salida (puerto A), generen +12V y un 0 logico (+0V) en los pines del puerto de salida, generen -12V. en otras palabras, la interface convierte +5V en la entrada en +12V en la salida y 0V en la entrada en -12V en la salida.
con lo anterior se ve claro que una secuencia de números en el puerto A {a,6,5,9,a,6,5,9,a,6,5,9...} entregarían al motor de pasos los pulsos correspondientes para hacerlo girar en una dirección. y una secuencia invertida lo haría girar en dirección contraria {9,5,6,a,9,5,6,a...}.

```
inicio
        movb    #$0a,porta
        movb    #$09,porta
        movb    #$05,porta
        movb    #$06,porta
        bra     inicio
```

Velocidad de movimiento: frecuencia de pulsos

Una sucesión de pulsos en un orden determinado hace al motor girar en una u otra dirección, detener la entrega de estos pulsos detendría al motor, sin embargo el microcontrolador es capaz de entregar cada numero a una velocidad que el motor no podría mecánicamente seguir puesto que la velocidad de ciclo del microcontrolador en de .125 μseg y la inercia angular del rotor incluyendo la carga (uniones y demás partes del robot que estén sostenidas) no permite un movimiento tan rápido. Por tanto es necesario cuidar la frecuencia de pulsos a los cuales el motor puede operar sin problemas.
Este problema se resuelve haciendo que el microcontrolador "desperdicie" ciclos de operación entre un paso y otro para esperar al motor a que realice el movimiento del paso anterior antes de entregar los pulsos del paso siguiente.

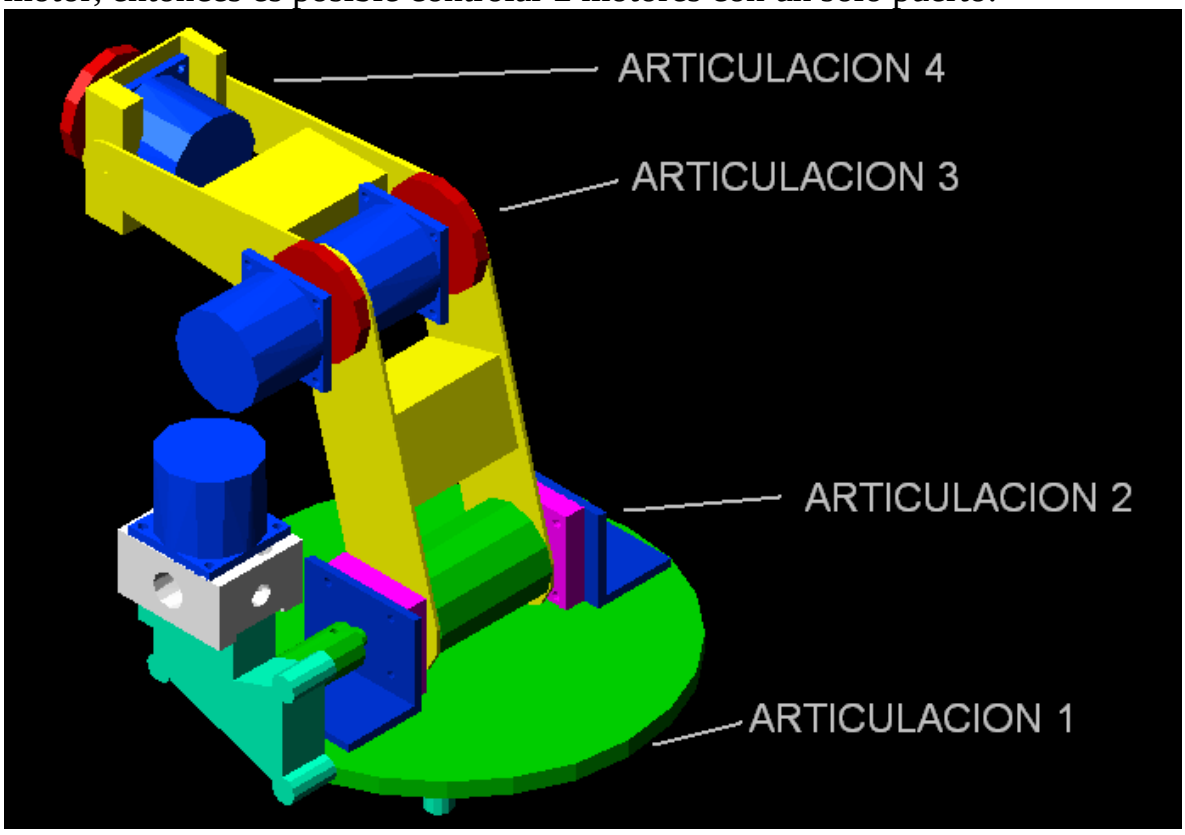
```
        #include        hc12.inc
        org     $900
frec    ds.b    1
        org     $800
;salida digital
        movb    #$ff,ddra
        movb    #$ff,ddrb
;-------------------------------------
        movb    #$ff,frec

inicio
        movb    #$0a,porta
        jsr     pausa
        movb    #$09,porta
        jsr     pausa
        movb    #$05,porta
        jsr     pausa
        movb    #$06,porta
        jsr     pausa
        bra     inicio
;pausa-------------------------------
pausa   ldx     frec
pausa1  dex
        bne     pausa1
        rts
```

El numero de ciclo que el programa desperdicie entre un paso y otro es el periodo de la frecuencia de pasos para el motor. el periodo que provoca una frecuencia de saturación en estos motores fue 11 ciclos entre paso y paso. Por tanto frecuencias mayores o con periodos mas chicos provocan una saturación que no permite girar al motor.

periodo en exadecimal	corriente (Amp)
c	.66
d	.40
e	.28
f	.23
10	.25
11	.27
12	.32
13	.38
14	.46
15	.56
16	.67
17	.76
18	.83
19	.91
1a	1.01
1b	1.00

1c	.18
1d	1.24
1e	1.33
1f	1.40
20	1.48
30	1.89
31	1.91
32	1.91
33	1.92
40	1.80
50	2.17
60	2.24
70	2.49
>ff	3.0

La programación para el control de un sistema mecatrónico como un robot implica necesariamente el manejo de varios grados de libertad a la ves o en este caso entregar los pulsos de varios motores de pasos a la ves. considerando que cada puerto en la salida del microcontrolador es capas de entregar valores binarios de 8 bits y que solo 4 bits son necesarios para el control de un solo motor, entonces es posible controlar 2 motores con un solo puerto.



Para esto hay que escribir un algoritmo que pueda afectar solo 4 bits de los 8 bits disponibles.
ejemplo:
supongamos que el puerto A controla a los motores 1 y 2 entonces una sucesión de movimientos en el motor 1 sin afectar al motor 2 se vería así:
puerto A = $aa
puerto A = $6a
puerto A = $5a
puerto A = $9a
en tanto que un movimiento siguiente del motor 2 sin afectar al motor 1 se vería así:
puerto A = $9a
puerto A = $96
puerto A = $95
puerto A = $99
Esto se logra aplicando una operación lógica AND en los bits del numero representado en el puerto A:
puerto A = %1010 1010 ($aa)
másacara= %0000 1111 ($0f)
puerto A -> puerto A AND mascara
recordando que 1 AND 1 = 1, 1 AND 0 =0, 0 AND 1 = 0 y 0 AND 0 = 0
obtenemos que el puerto A quedaría con el valor
puerto A = %0000 1010 ($0a)
con lo que los bits 7,6,5 y 4 quedaron en valor cero
Para introducir el nuevo valor del paso ($6) se usa una operación lógica OR
puerto A =%0000 1010 ($0a)
paso = %0110 0000 ($60)
puerto A -> puerto A OR paso
Con lo que queda en el puerto A
puerto A = $6a
Lo anterior puede aplicarse en cada caso para afectar a los bits correspondientes a un motor sin afectar a los demás pues es necesario mantener la señal de los pulsos que conforman el paso del motor mientras este esta quieto para que cada motor pueda ofrecer torque de torsión mientras que solo uno de ellos se mueve.
para este proyecto se dispuso que las salidas del puerto A controlen los motores 1 y 2; y las salidas del puerto B los motores 3 y 4. la distribución de pulsos para cada motor esta administrada por un parámetro llamado "motor" que puede tener valores 1, 2, 3 o 4 para indicar la salida en la que debe ser entregado el pulso.

```
acumulador B -> motor
acumulador A -> puerto A
si acumulador B=$1 ir a rutina M1
si acumulador B=$2 ir a rutina M2
acumulador A -> puerto B
si acumulador B=$3 ir a rutina M1
si acumulador B=$4 ir a rutina M2
M1:
acumulador A-> acumulador A AND $f0
acumulador A-> acumulador A OR pulso
si acumulador B=$3 ir a rutina M3
puerto A -> acumulador A
M3:
puerto B -> acumulador A
M2:
acumulador A-> acumulador A AND $0f
pulso -> pulso * $10
acumulador A-> acumulador A OR pulso
si acumulador B=$4 ir a rutina M4
puerto A -> acumulador A
M4:
puerto B -> acumulador A
```

este algoritmo distribuye la información llamada "pulso" en el puerto correspondiente, dependiendo del parámetro motor. Es decir cuando una instrucción de movimiento haga referencia a "mover el motor 1" debe evaluarse el parámetro motor=1.

Lo anterior deja ver un nuevo problema: cada ves que se de una instrucción de movimiento a un motor que ya ha sido movido antes habrá que saber cual fue el ultimo paso que le fue entregado para poder entregarle el siguiente.

En la programación de robot las instrucciones de movimiento para cada motor hacen referencia a una posición angular, la cual el la posición en la cual debe quedar para que el efector final pueda señalar la ubicación que se desea. Esto se debe a que el posicionamiento del efector final esta formado por el conjunto de las coordenadas que forman la posición de todas las articulaciones del cuerpo de robot (brazo articulado para este caso). por ejemplo: para poner al efector final en un punto del volumen de trabajo podría estar señalado por las coordenadas motor1=23, motor2=65, motor3=12, motor4=97.

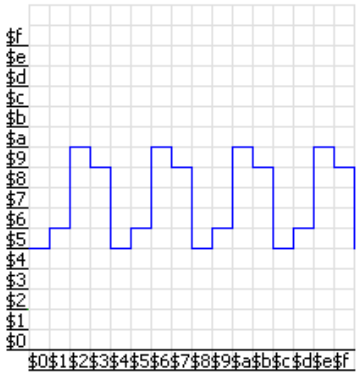
Por tanto las instrucciones de movimiento se escriben en términos de 3 parámetros básicos en la programación de movimiento para robot: numero de

articulación (motor), velocidad de movimiento (frecuencia), pocicion de la articulación (angulo final).

La posición de la articulación es un numero que depende de las características mecánicas del motor de pasos y de la transmisión, pues un paso en el motor hace que este se mueva una fracción de vuelta que esta determinada por el motor mismo, por lo que cualquier posición estará dada en términos de esta unidad mínima de posición. por ejemplo para mover al motor de la posición 30 a la posición 35 habrá que entregarle 5 pasos hacia a delante.

Sin embargo los pasos del motor están representados por números que se ciclan cada 4 pasos {a,6,5,9,a,6,5,9,a,6,5,9...} y la posición del motor esta dada por un numero entero que puede ser tan grande como lejos este la posición final del punto de referencia (HOME). La necesidad de tener una relación entre la posición del motor y el ciclo de pasos no lleva al siguiente análisis:
suponga una función dependiente de la posición final (s0) tal que

$$f(s0)=\begin{cases} \$5 \text{ para } s0=0 \\ \$6 \text{ para } s0=1 \\ \$a \text{ para } s0=2 \\ \$9 \text{ para } s0=3 \end{cases}$$



y que además
f(s0)=f(s0+4)
o también
f(s0)=(s0+n*4)
donde "n" es cualquier numero entero

si observamos el valor de la posición "s0" en numeración binaria se ve claro que los dígitos de las magnitudes 2° y 2' se ciclan cada 4 unidades

0	00 00
1	00 01
2	00 10
3	00 11
4	01 00
5	01 01
6	01 10
7	01 11
8	10 00

9 10 01
10 10 10
11 10 11
12 11 00
13 11 01
14 11 10
15 11 11

lo que sugiere escribir una función en términos de las magnitudes 2° y 2' (bit 0 y bit 1) del valor s0 en binario

f(b1,b0)= 5 + 1*b0 + 5*b1 - 2*b0*b1

donde "b0" es el valor de la magnitud 2° de "s0" y "b1" es el valor de la magnitud 2' de "s0" lo cual asegura que la función este ciclada cada 4 unidades. por ejemplo si s0=0 (0=%00)

f(0,0)= 5 + 1*[0]+ 5*[0] - 2*[0]*[0]= 5+5+0+0=5 ($5)

para s0=1 (1=%01)

f(0,1)= 5 + 1*[1]+ 5*[0] - 2*[1]*[0]= 5+1+0-0=6 ($6)

para s0=2 (2= %10)

f(1,0)= 5 + 1*[0]+ 5*[1] - 2*[0]*[1]= 5+0+5-0=10 ($a)

para s0=3 (3=%11)

f(1,1)= 5 + 1*[1]+ 5*[1] - 2*[1]*[1]= 5+1+5-2=9 ($9)

para s0=4 (4=%00)

f(0,0)= 5 + 1*[0]+ 5*[0] - 2*[0]*[0]= 5+5+0+0=5 ($5)

esta función construye el pulso correspondiente al paso del motor apartar de un numero dado. dejando ese numero en términos de la posición deseada para un motor determinado se puede escribir una rutina que entregue el pulso para cada una de las posiciones que existen entre la posición inicial y final.

Con lo anterior puede escribirse un programa que administre parámetros de instrucciones de movimiento para entregar pulsos en los puertos A y B declarados como salidas.

```
;[basico].....................................................................
        #include hc12.inc
        org    $9f0
s0      ds.b   1
z0      ds.b   1
z1      ds.b   1
z2      ds.b   1
z3      ds.b   1
z4      ds.b   1
motor   ds.b   1
```

```
pulso ds.b   1
PA    ds.b   1
PB    ds.b   1
periodo      ds.b   2
      org    $800
;salida digital----------------------------------------------------+
      movb #$ff,DDRA  ;SALIDA
      movb #$ff,DDRB  ; SALIDA
;valores iniciales-------------------------------------------------+
      movb #$0,z0
      movb #$0,z1
      movb #$0,z2
      movb #$0,z3
      movb #$0,z4
      movw #$afff,periodo
;rutina de prueba_____________________________________________
main  movb #$00,s0
      movb #$1,motor
      jsr    mover
      movb #$30,s0
      movb #$2,motor
      jsr    mover
      movb #$7f,s0
      movb #$1,motor
      jsr    mover
      movb #$31,s0
      movb #$2,motor
      jsr    mover
      bra    main
;_____________________________________________________________
;rutina de movimiento inicio----------------------------------------+
moverldaa  motor
      movb z1,z0
      cmpa #$1
      beq    pulsar
      movb z2,z0
      cmpa #$2
      beq    pulsar
      movb z3,z0
      cmpa #$3
```

```
        beq    pulsar
        movb   z4,z0
        cmpa   #$4
        beq    pulsar
;rutina de movimiento continuo----------------------------------------+
pulsar  ldaa   z0
        cmpa   s0      ;.....................(!)
        lbeq   fin     ;salir si es =
        bpl    retro   ;retro si es z0>s0
        inc    z0
        bra    CICLO
retro   dec    z0
;rutina de formacion del pulso correspondiente-------------------------+
CICLO   ldaa   #$5     ;inicio de la funcion f(b1,b2)
        brclr  z0,#%01,ciclo2
        inca           ;a=a+1
ciclo2  brclr  z0,#%10,ciclo3
        adda   #$5     ;a=a+5
        brclr  z0,#%01,ciclo3
        suba   #$2     ;a=a-2
ciclo3  staa   pulso   ;ya esta formado el pulso
        ldx    periodo
ret1    dex
        cpx    #$0
        bne    ret1
;entrega del pulso-----------------------------------
puerto  ldaa   PA      ;decicion del motor
        ldab   motor   ;repetir para el puertoB
        cmpb   #$1
        beq    M1      ;go to M1
        cmpb   #$2
        beq    M2      ;go to M2
        ldaa   PB
        cmpb   #$3
        beq    M1
        cmpb   #$4
        beq    M2
M1      anda   #$f0    ;m=1
        oraa   pulso
        cmpb   #$3
```

```
        beq   M3
        movb  z0,z1
        bra   Afinal
M3      movb  z0,z3
        bra   Bfinal
M2      anda  #$0f   ;m=2
        lsl   pulso
        lsl   pulso
        lsl   pulso
        lsl   pulso
        oraa  pulso
        cmpb  #$4
        beq   M4
        movb  z0,z2
        bra   Afinal
M4      movb  z0,z4
        bra   Bfinal
;-------------------------------------------------------------
Afinal staa   PA
        movb  PA,PORTA
        lbra  pulsar
Bfinal staa   PB
        movb  PB,PORTB
        lbra  pulsar
fin     rts
;_____________________________________________________________
```

Con el programa anterior es posible controlar un robot desde un micro controlador M68HC12B.